CPSC 449: Assignment 5

Fall 2014

Due: Wednesday, December 3, 2014 at **12:00pm noon**

1. [25%] Write a Prolog program for `substitute(X, Y, L1, L2)`, where list `L2` is the result of substituting `Y` for all occurrences of `X` in list `L1`. For example, the following is true:

   ```
   substitute(a,x,[a,b,a,c],[x,b,x,c])
   ```

   whereas the following is false:

   ```
   substitute(a,x,[a,b,a,c],[a,b,x,c])
   ```

2. [25%] Implement `select_pair/4`, which behaves in a way analogous to the built-in predicate `select/3`. While `select/3` generates all possible members of a list, `select_pair/4` generates all possible pairs of members of a list. Your implementation must generate the following results.

   (a) The query

   ```
   select_pair(X, Y, [1,2,3], Zs).
   ```

   should generate the following results (order is not important):

   ```
   X = 1, Y = 2, Zs = [3] ;
   X = 1, Y = 3, Zs = [2] ;
   X = 2, Y = 1, Zs = [3] ;
   X = 2, Y = 3, Zs = [1] ;
   X = 3, Y = 1, Zs = [2] ;
   X = 3, Y = 2, Zs = [3]
   ```

   (b) The query

   ```
   select_pair(1, 2, Xs, [3]).
   ```

   should generate the following results (order is not important):

   ```
   Xs = [1, 2, 3] ;
   Xs = [2, 1, 3] ;
   Xs = [1, 3, 2] ;
   Xs = [2, 3, 1] ;
   Xs = [3, 1, 2] ;
   Xs = [3, 2, 1] ;
   ```

3. [25%] Write a Prolog program to solve the following logic puzzle. There are five houses, each of a different color and inhabited by a man of a different nationality, with a different pet, drink, and brand of cigarettes.

(a) The Englishman lives in the red house.

(b) The Spaniard owns the dog.

(c) Coffee is drunk in the green house.

(d) The Ukrainian drinks tea.

(e) The green house is immediately to the right (your right) of the ivory house.

(f) The winston smoker owns snails.

(g) Kools are smoked in the yellow house.

(h) Milk is drunk in the middle house.

(i) The Norwegian lives in the first house on the left.

(j) The man who smokes Chesterfields lives in the house next to the man with the fox.

(k) Kools are smoked in the house next to the house where the horse is kept.

(l) The Lucky Strike smoker drinks orange juice.

(m) The Japanese smokes Parliaments.

(n) The Norwegian lives next to the blue house.

Who owns the Zebra? Who drinks water?

Hint: A naive generate-and-test solution is worth 20%. An incremental generate-and-test (i.e., with pruning) is worth the full 25%. Carefully document your data structures and algorithm. Failure to do so will cost you up to 8% (out of the 25% for this question).

4. [25%] Arithmetic expressions are defined by the following CFG:

```
Expr  ::=  lit(i)
       |   add(Expr, Expr)
       |   sub(Expr, Expr)
```

where i is an integer constant.

(a) [7%] Write a typing predicate `expr(E)` that succeeds whenever `E` is a syntactically correct arithmetic expression.

(b) [18%] Recall the "right-bracketed expressions" example in **[Thompson]** Section 14.2 (i.e., the `assoc` function discussed in set of lecture slides entitled "Recursive Algebraic Types, posted in week 5). Write a Prolog predicate `assoc(E1, E2)` that succeeds whenever `E2` is the right-bracketed form of `E1`. **Hint:** While the Haskell function `assoc` can rely on the ordering of pattern matching to ensure unique matching, your Prolog predicate `assoc` cannot rely on rule ordering. Instead, your rules should take care of ***disjoint*** cases of expressions. The following test can reveal if you are doing it right. Suppose you have the following fact in the database:

```
map(test, add(add(add(lit(2), lit(3)), lit(4)), lit(5))).
```

Then the following query

```
?- map(test, Ein), assoc(Ein, Eout).
```

should produce exactly one solution:

```
Ein = add(add(add(lit(2), lit(3)), lit(4)), lit(5)),
Eout = add(lit(2), add(lit(3), add(lit(4), lit(5))))
```

Failure to adhere to this guideline will result in a deduction of 5% (out of the 25% for this question).